## 9 Concurrent and Distributed Systems (RNW)

Sun's Network File System (NFS) is the standard distributed file system used with UNIX, and has gone through a progression of versions (2, 3, 4) that have gradually improved performance and semantics.

(*a*) Remote procedure call (RPC)

(*i*) Explain how Sun RPC handle byte order (endianness). [2 marks]

(*ii*) This approach may result in unnecessary work. State when this occurs and how might this be avoided. [2 marks]

(*b*) Network File System version 2 (NFSv2) and version 3 (NFSv3)

(*i*) A key design premise for NFS was that the server be "stateless" with respect to the client. State what this means for distributed file locking in NFSv2 and NFSv3. [2 marks]

(*ii*) Another key design premise for NFSv2 was the "idempotence" of RPCs; what does this mean? [2 marks]

(*iii*) One key improvement in NFSv3 was the addition of the READDIRPLUS RPC. Explain why did this helps performance. [4 marks]

(*iv*) NFSv3 implements what is termed "close-to-open consistency" for file data caching: if client C1 writes to a file, closes the file, and client C2 now opens the file for read, then it must see the results of all writes issued by C1 prior to close. However, if C2 opens the file before C1 has closed it, then C2 may see some, all, or none of the writes issued by C1 (and in arbitrary order). Close-to-open consistency is achieved through careful use of synchronous RPC semantics, combined with file timestamp information piggybacked onto server replies on all RPCs that operate on files.

Explain how close-to-open consistency allows performance to be improved. [4 marks]

(*v*) NFSv3 adds a new RPC, ACCESS, allowing the client to delegate access control checks at file-open time to the server, rather than performing them on the client. This allows client and server security models to differ.

Explain how this addition also helps implement close-to-open consistency in the presence of read caching. [4 marks]